LoRaSync: energy efficient synchronization for scalable LoRaWAN

Laurent Chasserat, Nicola Accettura, Pascal Berthou

LoRaSync: energy efficient synchronization for scalable LoRaWAN

Laurent Chasserat, Nicola Accettura and Pascal Berthou
LAAS-CNRS, Université de Toulouse, CNRS, UPS, Toulouse, France
Email: {firstname.lastname}@laas.fr

***Abstract*—Low-Power Wide-Area Networks (LPWAN) connect a large number of battery-powered wireless devices over long distances. Among them, Long Range Wide Area Networks (LoRaWAN) implement a Pure ALOHA medium access scheme to save device energy by minimizing the radio usage. However, frame collisions restrain the network scalability when the traffic load increases. In this context, synchronization can be used to exploit the available bandwidth more efficiently by controlling the timing of frame transmissions and reducing the collision probability. Such strategy allows to increase the network throughput at the cost of an extra energy demand due to the inherent overhead. In that, the entailed network scenario is still supposed to address low power applications. Therefore this paper timely presents *LoRaSync*, an energy-efficient synchronization scheme designed for LoRa networks of any size. An accurate clock drift model was established based on measurements made on real cheap devices, and leveraged to support the design of *LoRaSync*. Our mechanism has been used to evaluate the same ALOHA-based random access but on a time-slotted basis, thus increasing the maximum achievable throughput compared to the legacy access. Throughput and energy efficiency models are established to evaluate the performances of a *LoRaSync*-operated network. These models are validated with a simulation environment mimicking large-scale deployments, and then used to determine the most energy efficient slot size for any traffic load. As a final proof of concept, *LoRaSync* has been implemented and tested on a LoRa testbed to demonstrate the feasibility of our solution on real hardware.**

***Index Terms*—LoRaWAN, LPWAN, MAC Protocols, Energy Efficiency, Scalability, Machine-to-Machine Communications, Ad-Hoc Networks, IoT**

I. Introduction

By providing cheap devices with low power and long range communication capabilities [1], Low Power Wide Area Networks (LPWAN) keep gaining relevance in industrial and research environments. In particular, Long Range Wide Area Networks (LoRaWANs) have emerged as a very promising technology to support a wide range of distributed sensing applications [2]. The increasing availability of cheap wireless devices and the consequent growth of communication traffic have raised strong concerns regarding the scaling capabilities of LoRa deployments [3]–[9].

The reason behind LoRaWAN's scalability limitations lies in the simplicity of its medium access scheme. Indeed, to send information through the Internet, a node forges and immediately transmits a link layer frame without checking if the radio channel is free. This simple access scheme is the very well-known Pure ALOHA MAC (Medium Access Control) protocol [10]. Herein the channel throughput is limited to a maximum of 18% of the available bandwidth due to frame collisions.

Given that this strategy insures very poor network performances for high traffic loads, synchronization mechanisms have been explored as means to reduce the frame collision probability and increase the maximum LoRaWAN throughput [11], [12]. As a matter of fact, leveraging time synchronous slots to setup a very simple Slotted ALOHA random access scheme [13] doubles the maximum achievable throughput compared to its asynchronous version (*i.e.*, Pure ALOHA). More interestingly, a synchronization mechanism allows the development of even more sophisticated MAC layer schemes capable of handling various traffic requirements. For instance, a scheduled access could be designed to insure even collision-free communications. In order to share a common time reference across the network, all devices need to periodically receive timing information to cope with the natural drift of their embedded clock. To do so, they need to frequently switch on their radio to listen to incoming frames. This technique consumes a considerable amount of the available energy in the feeding batteries. Therefore, energy efficiency should be prioritized when designing the synchronization mechanism.

With this approach in mind, we introduced *Class S* in a previous contribution [14] as an extension of the LoRaWAN *Class B*, leveraging a beacon-based synchronization scheme to setup uplink transmission timeslots. Simulations showed that this solution could enable a slotted access over LoRa. However, a beacon skipping mechanism was required to limit the device radio usage and ensure energy efficiency. In this sense, we timely present the *LoRaSync* synchronization scheme, as a means to implement *Class S* on real cheap low power devices. More specifically, we measured the clock skew of typical LoRa hardware to evaluate how fast devices drift away from a given time reference. Based on this preliminary experimental evaluation, we designed *LoRaSync* with slots large enough to cope with clock errors. In that, the slot size and the maximum clock drift are used to compute the maximum amount of beacons that may be safely skipped while keeping devices synchronized. Such a beacon skipping mechanism is crucial to minimize the power consumption due to frame receptions, and eventually maximize the lifetime of feeding batteries.

In order to assess the performances of large-scale *LoRaSync*-enabled networks, throughput and energy efficiency models have been established. These models notably account

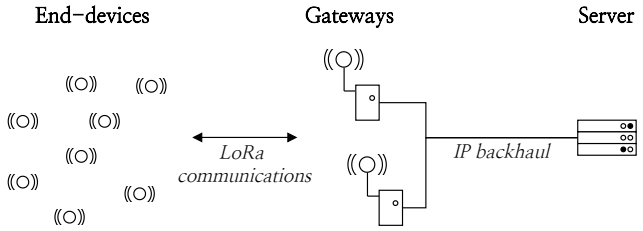
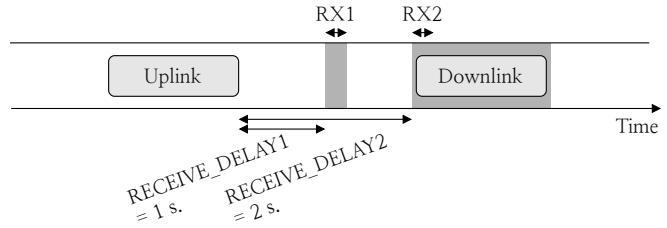


Fig. 1: LoRaWAN topology

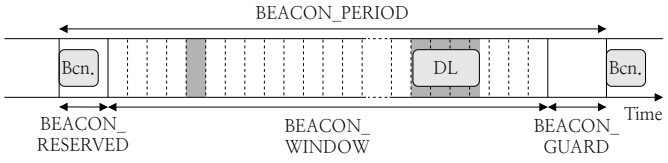
for slot size variations, reception slot enlargements and the proposed beacon skipping mechanism. They additionally have been validated through the LoRaWAN-sim simulation environment. Furthermore, the impact of the *LoRaSync* slot size on the overall energy efficiency of the network has been evaluated. Remarkably, the slot size has a twofold impact on the network behavior. On the one hand, increasing the slot length reduces the number of slots that fit into the slotframe, thus reducing the maximum achievable throughput. On the other hand, using large slots also allows devices to stay synchronized for longer time periods without receiving beacons, thus reducing their overall power consumption. As a consequence, we discovered that setting up the proper slot size value is functional to fit the best trade-off between throughput and power consumption. We therefore provide a methodology to determine the most energy efficient slot size for any traffic load.

In order to prove the soundness of the conceived *LoRaSync* mechanism, a real-world testbed has been setup to implement our synchronization scheme. This proof-of-concept demonstrates that the timing error can successfully be controlled despite the low-quality clocks found on such devices. In addition, this implementation allows us to highlight that the capture effect impacts the real-world throughput. We have therefore studied this phenomenon in more detail in a separate contribution [15].

From these premises, this contribution is organized as follows. First, technical background on the LoRa technology and the LoRaWAN MAC layer is provided in Section II. Section III describes related works about access scheme improvements for LoRaWAN. It justifies the need for synchronization to enhance the network performances, and highlights the novelty of our beacon-skipping approach that has been for the first time implemented on real hardware. Then, Section IV introduces the *LoRaSync* design, based on real clock skew measurements. Section V presents and validates models for the throughput, power consumption and energy efficiency in *LoRaSync*-enabled networks, that are then leveraged to determine the most energy efficient slot sizes. For the sake of completeness, a preliminary evaluation of *LoRaSync* in real LoRa networks is presented in Section VI through the description of a proof-of-concept implementation. Finally, Section VII discloses concluding remarks and research perspectives.



(a) *Class A*



(b) *Class B*

(c) *Class C*

Fig. 2: Access schemes defined by the different LoRaWAN classes

II. Background on LoRa and LoRaWAN

The LoRa LPWAN technology is well known for its capability to gather data over wide deployment areas. The term LoRa specifically designates the physical layer, while LoRaWAN is the Medium Access Control (MAC) layer that was designed on the top of it. The proprietary LoRa modulation relies on a Chirp Spread Spectrum (CSS) technique, which has proven to be resilient to multi-path interference and channel fading [16]. It allows transmissions to reach up to 10 km ranges under ideal conditions [3], with a relatively low power operation. A key parameter of this technique is the Spreading Factor (SF), that can be associated to the chirp sweeping time [17]. A bigger SF results in a longer transmission range, but also in a bigger Time on Air (ToA). High SFs are therefore featured with a low data rate and high energy consumption. Later in this contribution, we will focus on using the smallest SF (SF7) because it results in the shortest ToA, and therefore in the minimal energy consumption.

The LoRa physical layer modulation is proprietary, however the upper layers are open and well documented [18]. In particular, the LoRa Alliance provides a common view on the MAC layer implementation through the LoRaWAN specification. This standard presents a network topology in which nodes communicate with gateways via LoRa communications, and gateways interact with servers through an IP backhaul. This architecture is represented in Figure 1.

LoRaWAN defines several device classes with specific communication modes able to fit different types of applications. *Class A* is the default scheme that all devices must implement. In this mode, the uplink (device to server) transmissions are

Synchronization mechanism	Synchronization strategy	In-band synchronization	Scalable to any number of devices	Real-hardware implementation	LoRaWAN compliant
Piyare *et al.* [19]	Low-power wake-up receivers		x	x	
Beltramelli *et al.* [20]	FM-RDS signals		x	x	
Polonelli *et al.* [11]	Acknowledgements	x		x	x
Haxhibeqiri, Garrido *et al.* [12] [21]	Acknowledgements			x	x
Singh *et al.* [22]	Acknowledgements	x		x	x
Abdelfadeel *et al.* [23]	Beaconing	x	x		
Reynders *et al.* [24]	Beaconing	x	x		
Zorbas *et al.* [25]	Beaconing	x	x	x	x
Tessaro *et al.* [26]	Beaconing	x	x	x	
LoRaSync	Beaconing	x	x	x	x

TABLE I: Comparison of related works about LoRa synchronization

carried-out in a Pure ALOHA manner, and followed by two 30 ms reception slots (*i.e.*, RX1 and RX2), providing the server with an opportunity to return a downlink message. The reception slot length corresponds to the time a device needs to detect a LoRa preamble. When such a preamble is identified, the device maintains its radio in a listening state until the end of the transmission in order to demodulate the data symbols. This behavior is depicted in Figure 2a, where a downlink frame is received during the second reception slot RX2. When the server has a pending message for a *Class A* node, it thus has to wait until this node sends an uplink message to have a downlink transmission opportunity. As a result, *Class B* was introduced to reduce and bound the downlink delay. *Class B* devices operate like *Class A* ones for uplink transmissions, but they also periodically open additional reception slots, offering frequent downlink opportunities. To do so, they need to synchronize to the network by listening to beacons broadcast by the gateways, as shown in Figure 2b. Finally, *Class C* devices (*cf.* Figure 2c) must continuously listen to possible incoming frames when they are not transmitting. This mode obviously consumes a significant amount of power and is reserved for certain critical actuators.

III. Related works

A LoRaWAN network relies on gateways to gather frames transmitted by large numbers of low-power end devices and forward them to a data collection server. The research community has raised strong concerns regarding the scalability of such a topology under a pure ALOHA access [3]–[9]. Indeed, frame collisions experienced with this simplistic access scheme result in a poor network capacity. In this Section we explore the alternative access strategies that have been proposed by the research community to enhance the performances of the LoRa MAC layer.

Aside from synchronization, a common MAC strategy to increase the network capacity is the use of Carrier Sense Multiple Access (CSMA) schemes [27]. Approaches based on CSMA rely on listen-before-talk features to alleviate collisions. However their application to LoRa communications is still stammering in the current state-of-the-art. First of all, LoRa transceivers prior to the SX126x generation were only capable of detecting preamble symbols transmitted at the beginning of a transmission, and not the data symbols that compose the rest of the frame. After that, Semtech implemented a Channel Activity Detection (CAD) feature [28] allowing them to detect the presence of data symbols as well. This improvement suggests that a Clear Channel Assessment (CCA) procedure, crucial requirement to the conception of a CSMA access, could be implemented. However it was found in [29] that this CCA quickly becomes unreliable when the distance between the concurring nodes increases. Despite of this weakness, several listen-before-talk schemes have been proposed [30]–[33] to enhance the network performances. Yet, their application to real-life networks where a large number of widely spread nodes are in competition remains uncertain. Indeed if no reliable sensing can be performed passed a certain distance, it is expected that the hidden terminal problem [34] will drastically weigh on the protocol performances. For this reason, we believe that access schemes relying on synchronization are more promising. In fact, even a very simple Slotted ALOHA access as the one explored in this paper nearly doubles the network capacity with very little impact on the end-to-end delay. Furthermore, synchronization will later allow to setup scheduling algorithms performing even better than CSMA strategies, taking advantage of the centralized server already present in the LoRaWAN topology. In the rest of this Section, we therefore focus on contributions that leverage a synchronization strategy to exploit the available bandwidth more efficiently.

A first approach to seamlessly synchronize devices without impacting the LoRa communications is the use of out-of-band communications. In [19], Piyare *et. al.* propose an on-demand Time Division Multiple Access (TDMA) scheme that leverages another low-power communication technology (*i.e.*, micro-Watt wake-up receivers) to activate all devices and provide them with a common time reference before starting LoRa data exchanges. Similarly, Beltramelli *et. al.* [20] propose the use of FM-RDS signals to achieve synchronization in LoRa networks. The main drawback of these strategies is that they need LoRa devices to be equipped with additional hardware components, thus making such solutions unusable with current products available on the market.

Another popular strategy is to use individual acknowledgments to synchronize devices upon demand. For instance in [11], Polonelli *et. al.* implemented a Slotted ALOHA scheme on real hardware, using *Class A* reception windows to share

time information through acknowledgments. In [12], the synchronization frame additionally carries a timeslot assignment schedule to organize the transmission of all devices based on their traffic needs. Notably, Bloom filters are used to reduce the size of this extended downlink frame and enhance the efficiency of this mechanism. This work is then leveraged in [21] to setup a transmission scheduling protocol for LoRaWAN. These contributions account for the clock drift of devices and are supported by real-hardware implementations. This shows the feasibility of the strategy, but also reveals that a re-synchronization is periodically required. On this regard, the gateway's Duty Cycle limitations bound the network size that can be handled by this mechanism [35]. Besides, most LoRa gateways are half-duplex [36], meaning that they cannot transmit and receive frames at the same time. In that, the downlink traffic will considerably increase if many nodes require an ack-based synchronization. The gateway will therefore be unreachable during the transmission of these messages, which will eventually affect the uplink throughput. Another proof-of-concept implementation of an ack-based synchronization has been proposed in [22], and leveraged to setup a channel-hopping scheme. However the authors focus on using a single slave node, and therefore do not realize the scaling difficulties presented above.

These issues are avoided when using beacons to advertise the network and synchronize end devices. Such an approach scales with any network size, since the reception of a beacon frame can serve as time source for all devices listening to it on the air. That is why we chose the LoRaWAN *Class B* beaconing mechanism as a basis to design *Class S* [14], an access scheme allowing slotted uplink communications. One of the key conclusions of this contribution was that a beacon-skipping mechanism was highly desired in order to save battery on the low-power LoRa devices. Several contributions [23]–[26] have explored beacon-based synchronization, but until now the idea to fine-tune the beacon reception periodicity in order to foster energy efficiency hasn't been explored in depth. These other approaches are detailed below, and for each we state which key differences have been identified in regards to *LoRaSync*. In [23], beacons are leveraged to share the settings of a fine-grained scheduling algorithm designed for bulk data collection. The problem of bulk data collection differs from the continuous traffic management studied in our paper, as it targets networks in which the base-station is not always available. In that, it corresponds to use-cases in which gateways are carried by drones or satellites, and periodically hover over sensors deployed in very remote areas. Alternatively, a beaconing mechanism is used in [24] to group transmissions with similar RSSI and SF within the same timeslots. The objective of their paper is to reduce the impact of the capture effect, notably improving the network fairness. Their goal is therefore different from ours, because we rather focus on extending the network capacity while maximizing energy-efficiency. With TS-LoRa [25], Zorbas *et. al.* tackle the problem of autonomous slot assignment in synchronized LoRa networks with acknowledged traffic. SACK (synchronization/acknowledgement) packets are introduced to serve both as synchronization beacon and group acknowledgement. The viability of this protocol is asserted with a real-hardware implementation, yet the reception of SACK packets increases the energy consumption by 50%. Authors claim that they offer a better energy efficiency than legacy LoRaWAN, but in fact they compare their implementation to the LoRaWAN acknowledged mode in which devices already have to spend energy for ACK receptions. This approach can be put into question since, as explained above, it has been shown in [35] that bidirectional traffic heavily affects the uplink throughput. That is why we focus on unconfirmed traffic in the scope of this paper, which makes our approach completely different from TS-LoRa. Finally, beacon synchronization was once again used in [26] with a focus on industrial networks featured with periodic, predictable traffic. Clock drift measurements and corrections are notably emphasized in this last contribution. However the targeted protocol is an offline-scheduling algorithm designed for industrial sensor networks with predictable and periodic traffic. Conversely, LoRaSync is designed for general-purpose LoRa deployments with unpredictable traffic and possible layout changes.

In this landscape, *LoRaSync* is introduced as a scalable beaconing-based synchronization approach that focuses on energy efficiency thanks to its beacon-skipping mechanism. It is built on top of the LoRaWAN Class B beacon window structure, and can therefore be easily implemented on typical LoRa hardware with just a few firmware adjustments (*c.f.* Section VI-B). This beacon-skipping mechanism relies on real clock drift measurements in order to space out the synchronization events as much as possible without creating inter-slot transmission overlaps, thus keeping the device power consumption to the minimum. To support this claim, other synchronization schemes found in the literature are compared against *LoRaSync* in Table I. Its columns refer to the main features of the different synchronization strategies mentioned in this Section. In detail, we want the synchronization to be in-band so that additional hardware components do not need to be added to the LoRa devices. We also mark the acknowledgment-based approaches as non-scalable, because we have seen that the gateway's half-duplex characteristic and DC regulations limit their ability to handle a large number of devices without weighing on the uplink throughput. Moreover, we highlight the approaches that have demonstrated the feasibility of their solution on real-hardware. Finally, the papers relying on the existing LoRaWAN *Class A* transactions and on the *Class B* beacon window structure have been indicated as well. For these, implementing the proposed protocols in existing networks require less changes to the device firmware.

IV. LoRaSync requirements and design

With *LoRaSync*, we propose an energy-efficient and scalable synchronization approach tailored for LoRa deployments. It is designed to support the development of time-slotted access schemes for such networks, with an effort to minimize the device power consumption. This Section first provides an

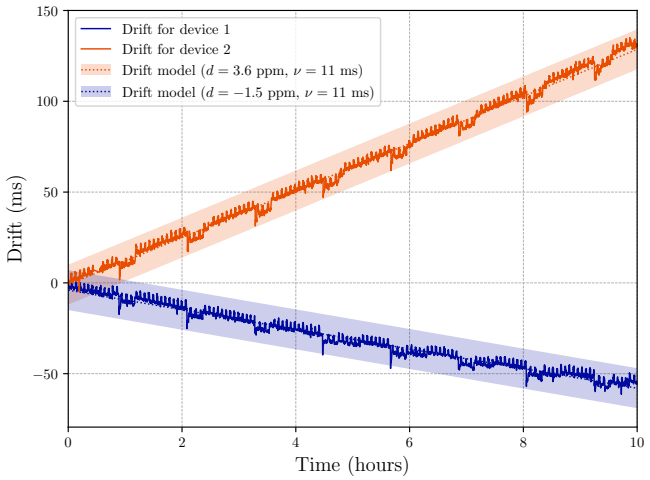


Fig. 3: Device drift measurements

experimental evaluation of the clock drift experienced with low-cost LoRa devices. Second, we detail the reasons why we recommend to parameterize all transmissions with SF7, and why we concentrate on using the largest payload length allowed in the scope of this paper. Based on these premises, we then present the *LoRaSync* design which includes *(i)* the synchronization strategy, *(ii)* a slotframe structure resilient to the device clock skew and *(iii)* a beacon-skipping mechanism designed to save battery power.

A. Device drift measurement and modeling

The *LoRaSync* synchronization mechanism aims at efficiently correcting the natural clock drift of cheap wireless devices to allow slotted uplink communications. Therefore, it is necessary to understand how this drift can be modeled before tackling the *LoRaSync* design. To do so, we use the testbed presented in Section VI, and program LoPy devices[1] to periodically transmit their local time. By comparing the intervals between the subsequent transmissions as seen by the gateway (which benefits from a very accurate GPS time), and the estimation of the same intervals done by the devices, we are able to evaluate the variations of the timing error. This clock skew has been plot in Figure 3 for two devices.

In most related works found in the literature [11], [20], [26], the clock drift is modeled as a linear function with the assumption that the sensor temperature is constant. This is referred to as the Simple Skew Model (SKM) [37], and it is verified in our experiment as we can see on Figure 3 that the device drift follows a linear trend on the long term. For cheap devices equipped with low-quality clock crystals, a 20 parts per million (ppm) worst-case drift coefficient is typically assumed.

However, it should also be noted that the clock skew displays a substantial noise around this overall trend, which must be estimated and is accounted for when designing the *LoRaSync* margins. This jitter has been mentioned in [38], but the authors chose not to include it in their modeling for simplicity reasons. Such a noise is indeed rarely considered in practice, even though for such cheap hardware it can result in significant errors for short time spans. As a result, we define a clock drift interval model in which d is the drift coefficient in ppm, and ν the maximum deviation from the linear model due to noise. Therefore, we are assured that a device featured with such parameters which hasn't been synchronized for a duration of t will experience a drift comprised within the interval $(d \cdot t) \pm \nu$. The value of ν has been experimentally obtained by observing the maximum deviation from the linear trend found in the samples collected by the two devices and rounding it to the higher value. Following this methodology, we find $\nu = 11$ milliseconds. Such a model has been plot next to the real drift on Figure 3 with d adapted to the drift coefficient of each of the two devices, respectively -1.5 and 3.6 ppm. This preliminary drift evaluation will later be used to fine tune the *LoRaSync* parameters. Besides, we note that the worst-case 20 ppm drift hypothesis is checked for the two considered devices.

B. Spreading Factor selection

We have seen in Section II that the SF is a key parameter of the LoRa modulation, which has an impact on the range and ToA of frame transmissions. In this part, we justify our choice to recommend using only the smallest SF, SF7.

First of all, using the other SFs is detrimental in terms of energy efficiency. In detail, a higher SF results in a lower data rate and longer ToA, the only benefit being the improved transmission range. The consequence of having a longer ToA is that devices will have to use their radio longer to transmit the same amount of data, and will thus consume a greater amount of energy. Indeed we know that given SF the transmission Spreading Factor and BW its bandwidth, the symbol duration T_{Sym} can be computed as such [39]:

$$T_{\text{Sym}} = \frac{2^{SF}}{BW} \tag{1}$$

As a result, the ToA magnitude grows proportionally to 2^{SF}, meaning that high SFs drastically affect the device's energy efficiency. In this paper we have designed LoRaSync on top of SF7, because it showcases the best performances in terms of both data rate and power consumption. Yet, the same designing logic proposed throughout this paper for SF7 can be applied to work with other SFs: for each additional SF, a new timeslot duration must be defined to accommodate the maximum frame size related to that SF. The resulting slotted structures (each on a separate SF) can herein coexist on a given channel.

SFs are presented as orthogonal by Semtech [39], meaning that transmissions occurring on the same channel but different SFs should be able to overlap without interference. However this claim has been put into question by the research community [40]. Specifically, Croce *et al.* measure a 16 dB co-channel inter-SF rejection threshold. Such a RSSI difference is very likely to occur in near-far conditions, which are commonly

[1]Pycom website: https://pycom.io/product/lopy4/

experienced in large scale LoRaWANs, especially if high SFs featured with long transmission ranges and frame lengths are used. Multi-SF LoRa deployments can therefore not be studied as the simple superposition of independent networks. This research finding lets us prefer deploying multi-gateway single-SF LoRaWANs to a single-gateway multiple-SF LoRaWAN, if the goal is to cover a wider geographical area. Yet, this hypothesis is out of scope for this paper and its validity and application scope will be investigated in future works. When the use of several non-orthogonal SFs is still the preferred option, the availability of a slotted structure for each SF (of course, featured by a SF-specific timeslot size) would be beneficial to the definition of access schemes (ranging from slotted Aloha to collision-free scheduling) able to bound possible inter-SF collisions. This is also a research topic to be investigated in future works.

C. Payload size standardization

In this contribution, the timeslot length is derived from the maximum frame ToA allowed in the network, $ToA_{\max}$ (*c.f.* Equation 4). This parameter depends on the SF and payload size. We consider the maximum payload size authorized by the LoRaWAN regional parameters [41] (*i.e.*, 250 bytes in Europe). LoRaSync can still be used with heterogeneous payload sizes, but optimal performances are obtained when every frame is featured with the maximum ToA.

The advantage of such a design in our situation is twofold. First, it allows us to use the available bandwidth more efficiently when a slotted access is utilized. Indeed, having frames with a length smaller than the maximum would create gaps between slotted transmissions, thus wasting a portion of the bandwidth. In the worst case, a slotted access could even showcase a lower throughput than Pure ALOHA. Conversely, an homogeneous frame length ensures the gaps in between frames are kept as small as possible, and guarantees that the bandwidth is used more efficiently.

Additionally, this hypothesis is also beneficial to the energy efficiency of devices. In fact when the maximum payload length is selected, the proportion of frame overhead compared to the amount of payload bytes is minimized. Therefore, a larger portion of the device energy is spent transmitting useful bytes, thus ameliorating the network energy efficiency from the application layer perspective. Such a requirement can be easily fulfilled by forcing devices to buffer their messages, and concatenate them into a single payload when the sufficient amount of data has to be accumulated. The detrimental aspect of such an approach is that the delay between frames increases because of such a buffering mechanism. However LoRa networks are typically not used for time-critical applications, so we consider this hypothesis to be reasonable.

In a nutshell, enforcing the transmission of frames with the maximum allowed size is beneficial to both extend the network capacity and foster device energy efficiency.

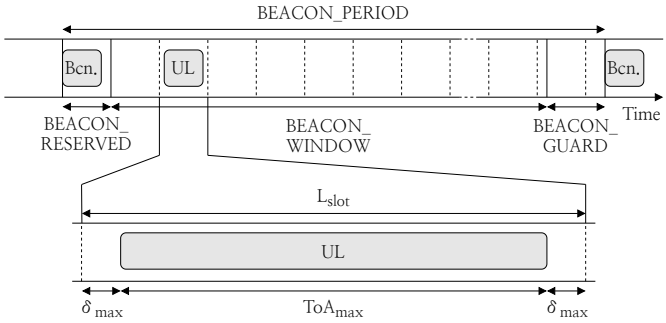


Fig. 4: LoRaSync beacon period and slot structure.

TABLE II: Slotframe intervals

Interval	Length
BEACON_PERIOD	128 s.
BEACON_RESERVED	2.12 s.
BEACON_WINDOW	122.88 s.
BEACON_GUARD	3 s.

D. LoRaSync design

Building a slotted access requires signaling messages to be exchanged in the network to share a common time reference among all devices. To do so, *LoRaSync* exploits the LoRaWAN *Class B* beaconing mechanism. Therefore, this scheme can run on typical LoRa hardware with a few adjustments to the firmware (*c.f.* Section VI-B). A synchronization event is defined as a tuple $(ts_{\text{bcn}},\ cnt_{\text{bcn}})$, where ts_{bcn} is the beacon's UTC timestamp and cnt_{bcn} is the device's internal counter value at the beginning of the beacon reception. Such event is saved by the device at each beacon reception. UTC timestamps are defined as the total number of microseconds elapsed since the Unix epoch. Device counters are incremented every microsecond as well, therefore it is possible to add and substract counter values to timestamps. Hence, the timestamp *ts* associated with any counter value *cnt* may be computed with the following equation:

$$ts(cnt) = ts_{\text{bcn}} + (cnt - cnt_{\text{bcn}}) \tag{2}$$

The precision of this estimator depends on both the internal clock quality and the time elapsed since the last synchronization event. Indeed, indicating the device's worst-case clock drift with d (expressed in parts per million, ppm) and ν the noise margin, the worst-case timing error *err* may be computed for any counter value *cnt* with the following equation:

$$err(cnt,\ d,\ \nu) = (cnt - cnt_{\text{bcn}}) \cdot d + \nu \tag{3}$$

This worst-case error is notably used to enlarge the width of the reception slot each time a device needs to receive a beacon. This feature allows the radio to be in a listening state when the beacon is sent, regardless of its current clock misalignment. *LoRaSync* implements *Class S* [14], meaning that the beacon window is divided into timeslots dedicated to uplink transmissions. As a means to preserve bidirectional communications, *Class S* transmissions are followed by the RX1 and RX2 reception slots just like *Class A* ones. In order to

adapt *Class S* to real hardware constraints, a maximum clock offset threshold δ_{max} is defined. The slot size computation thus accounts for δ_{max} and the maximum frame Time on Air (ToA) allowed:

$$L_{\text{slot}} = \textit{ToA}_{\text{max}} + 2 \cdot \delta_{\text{max}} \tag{4}$$

With such a slot size, a device may not trigger a transmission that overlaps over two different slots as long as the current clock skew of this device is smaller than δ_{max}. The associated slotframe layout within the *BEACON_PERIOD* is pictured in figure 4. It is divided in three intervals: *BEACON_RESERVED* which is dedicated to the beacon reception, *BEACON_WINDOW* in which slots are layed-out, with the last slot overlapping onto *BEACON_GUARD*. These intervals are identical to the ones used in the LoRaWAN *Class B* (*c.f.* Table II) in order to facilitate the implementation of *Class S*. According to the scheme pictured so far, the number of slots in the slotframe is:

$$n_{\text{slots}} = \left\lceil \frac{\textit{BEACON\_WINDOW}}{L_{\text{slot}}} \right\rceil \tag{5}$$

A beacon-skipping mechanism has been implemented in order to ameliorate the energy efficiency of the protocol. In particular, n_{skip} is defined as the maximum number of beacons that can be skipped by a device before it needs to get synchronized again. Given the worst-case drift coefficient d, the noise margin ν and the maximum offset allowed δ_{max}, n_{skip} is equal to the biggest $k \in \mathbb{N}$ that fits the following inequality:

$$(k+1) \cdot \textit{BEACON\_PERIOD} \cdot d + \nu \leq \delta_{\text{max}} \tag{6}$$

This means that increasing the maximum offset allowed (and thus the slot size) reduces the synchronization beacon periodicity, eventually decreasing the device power consumption induced by beacon receptions.

V. Modeling and optimization

This section presents the throughput, power consumption and energy efficiency models used to evaluate the network performances. For the sake of comparison, both *Class A* (*i.e.* Pure ALOHA) and a Slotted ALOHA scheme running on top of *LoRaSync* will be assessed. It has been shown that bidirectional traffic drastically weighs on the throughput of LoRa networks [35]. For this reason, transmissions do not require acknowledgments in the scope of this paper. It is also well-known that the capture effect occurs in LoRa deployments, and has a strong impact on the network throughput [4], [42]. Including this effect to the throughput model is not a trivial task. We therefore tackle this problem in a separate contribution [15], in which more advanced models are introduced and validated with real performance measurements performed on our testbed. In this paper, we therefore assume that overlapping frames are necessarily lost.

Let us first remind the most usual ALOHA throughput modeling with a finite number of devices n, as presented in [43]. The number of frames generated by any device during a duration t is modeled by a Poisson distribution of parameter λ. Besides, we make the hypothesis that all frames in the network have the same Time on Air (ToA) (*c.f.* Section IV-C) and do not require an acknowledgment. Additionally, Pure ALOHA devices do not use any kind of buffering, and slotted ones use a buffer of size 1 that simply allows them to wait for the start of the next slot. Taking this ToA as the elementary time unit of our modeling, we then define the probability p that a pure ALOHA device generates at least a frame during such a time unit. Since a buffer of size 1 is assumed, terminals ignore new packet arrivals while they are busy trying to transmit a packet. Therefore, p follows the cumulative distribution function of the exponential law [44]:

$$p = p[X \leq 1] = 1 - e^{-\lambda} \tag{7}$$

The probability that exactly k among n devices generate a frame during a time unit can then be derived with the binomial coefficient:

$$P[k \text{ in } n] = \frac{n!}{k!(n-k)!} p^k (1-p)^{n-k} \tag{8}$$

For Pure ALOHA, the average throughput T_p (in erlangs) is equal to the probability that exactly one device generates a frame during a time unit, and that none of the other $(n-1)$ devices do during the previous one:

$$T_p = P[1 \text{ in } n] \cdot P[0 \text{ in } (n-1)] = np(1-p)^{2(n-1)} \tag{9}$$

In order to adapt this modeling to *LoRaSync*, we introduce q as the probability that exactly one device generates a frame for the duration of a slot. We once again leverage the exponential law:

$$q = p\Big[X \leq \frac{L_{\text{slot}}}{\textit{ToA}_{\text{max}}}\Big] = 1 - e^{-\lambda \frac{L_{\text{slot}}}{\textit{ToA}_{\text{max}}}} \tag{10}$$

Besides, the k_{s} coefficient is introduced as a means to represent the actual transmission time available per slotframe:

$$k_{\text{s}} = \frac{n_{\text{slots}} \cdot \textit{ToA}_{\text{max}}}{\textit{BEACON\_PERIOD}} \tag{11}$$

Using once again the binomial coefficient to compute the probability that exactly one device generates a frame during the duration of a slot, the average slotted ALOHA throughput T_s (in erlangs) is equal to:

$$T_s = k_s n q (1-q)^{n-1} \tag{12}$$

In order to model the energy efficiency of the devices, their power consumption (P) must first be computed. This is done by considering its value in the transmission, reception and sleep states separately. We note P_{TX}, P_{RX} and P_{SLEEP} the power consumption of a typical SX1276 LoRa transceiver in transmission, reception and sleeping states respectively, when equipped with a 3.3V battery voltage [45]. It is computed with the transceiver's supply current in the considered state, respectively 20, 10.8 and $0.2 \cdot 10^{-3}$ mA. Our modeling accounts for the LoRaWAN 30 ms. reception slots following the uplink transmissions (*i.e.*, RX1 and RX2). For this purpose, we introduce the slot listening rate ρ_s:

$$\rho_s = \lambda \cdot \frac{2 \cdot 0.03}{\textit{ToA}_{\text{max}}} \tag{13}$$

The overall network power consumption therefore is:

$$P_p = \left[\lambda P_{\text{TX}} + \rho_s P_{\text{RX}} + (1 - \lambda - \rho_s) P_{\text{SLEEP}}\right] \cdot n \quad (14)$$

For the *Class S* model we need to additionally consider the time spent in a receiving state for beacon reception purposes. We define the device beacon reception period T_{bcn} as:

$$T_{\text{bcn}} = \textit{BEACON\_PERIOD} \cdot (n_{\text{skip}} + 1) \quad (15)$$

The beacon time on air is noted $\textit{ToA}_{\text{bcn}}$. With *LoRaSync*, the beacon reception window is enlarged to cope for the worst-case drift that can occur during T_{bcn} with a drift coefficient d and a noise margin ν. Indeed, the device transmission may be shifted by an offset ranging from $-d \cdot T_{\text{bcn}} - \nu$ to $d \cdot T_{\text{bcn}} + \nu$. At the end of the beacon reception, the device switches its radio to sleep mode, so that the elapsed listening time may range from $\textit{ToA}_{\text{bcn}}$ to $\textit{ToA}_{\text{bcn}} + 2 \cdot (d \cdot T_{\text{bcn}} + \nu)$ depending on the transmission offset. We assume that the drift coefficient d of all devices is uniformly distributed in the interval $\pm d$, therefore the average time spent in listening mode is $d \cdot T_{\text{bcn}} + \nu$. As a result, the beacon listening rate ρ_b is:

$$\rho_b = \frac{\textit{ToA}_{\text{bcn}} + d \cdot T_{\text{bcn}} + \nu}{T_{\text{bcn}}} \quad (16)$$

The power consumption model can then be expressed as such:

$$P_s = \left[(\rho_s + \rho_b) P_{\text{RX}} + \lambda P_{\text{TX}} + (1 - \rho_s - \rho_b - \lambda) P_{\text{SLEEP}}\right] \cdot n \quad (17)$$

The energy efficiency (E) is expressed in Bytes per Joule and can be obtained by dividing the throughput by the power consumption. The throughput must however be expressed in bytes per second first, which is achieved by the term $\frac{\textit{bytes}_{\text{pkt}}}{\textit{ToA}_{\text{pkt}}}$ in the equations below. The energy efficiency models for Pure and Slotted ALOHA, respectively E_p and E_s, are therefore:

$$E_p = \frac{T_p}{P_p} \cdot \frac{\textit{bytes}_{\text{pkt}}}{\textit{ToA}_{\text{pkt}}} \quad (18)$$

$$E_s = \frac{T_s}{P_s} \cdot \frac{\textit{bytes}_{\text{pkt}}}{\textit{ToA}_{\text{pkt}}} \quad (19)$$

A. Model validation and analysis

TABLE III: Simulation parameters

Parameter	Value
Simulation duration	24 hours
Number of seeds	10
Frame time-on-air	389.376 ms
Beacon time-on-air	173.06 ms
δ_{max}	2.56 to 53.76 ms
Data rate	DR5 (SF7 with bandwidth 125 kHz)
Frame generation rate	Varies from ~ 0.5 to ~ 2.5 pkts. / h.
Number of channels	1
Number of devices	2000
Device buffer size	1
n_{skip}	Fixed according to δ_{max}
Downlink data messages	Disabled
Acks and retransmissions	Disabled
Sensor voltage	3.3 V
Sensor current intensity	20 mA TX, 10.8 mA RX [45]

LoRaSync targets large-scale deployments that experience scalability issues with the legacy *Class A*. Given the difficulty to approach such a large scale scenario with a research testbed, the models presented above have been validated using the LoRaWAN-Sim simulation environment to instantiate a network of 2000 devices. This analysis is completed by a proof-of-concept implementation on a few devices (*c.f.* Section VI) to demonstrate the feasibility of our approach. The LoRaWAN *Class A* and a Slotted ALOHA access over *LoRaSync* have been replicated in the simulator. Additionally, LoRaWAN-Sim integrates a linear clock drift feature that allows us to replicate the LoRaSync clock correction. The SF has been set to 7 for the reasons stated in Section IV-B. All transmissions occur on a single channel, and are parameterized with a coding rate of $4/5$ with explicit header and cyclic redundancy check enabled. As specified in Section IV-C, all frames carry the maximum payload size allowed by the LoRaWAN specification (*i.e.* 250 bytes) to minimize the amount of overhead transmitted by the devices, which results in a ToA of 389.376 ms. *LoRaSync* was set up with a worst-case drift coefficient $d = 20$ ppm, which is a typical value for low-quality crystals found in such cheap hardware. Throughout this analysis, error bars represent 99% confidence intervals along the x and y axes computed with Student's t law. In order to facilitate the reproduction of our experiments, detailed simulation parameters are provided in Table III.

The simulated throughput for Pure and Slotted ALOHA are compared to the models in Figure 5, and the energy efficiency equivalent is displayed in Figure 6. In all cases, the model curves match the simulation results, showing the consistency between the modeling and the protocol implementation on LoRaWAN-Sim. In each plot *Class A* is compared to *LoRaSync* for several slot sizes. This comparison notably shows that the models faithfully capture the impact of the transmission margin size on the overall performances, which will be relevant for the analysis of Section V-B. Specifically, the chosen slot sizes correspond to the δ_{max} values of 2.56, 12.8, 28.16 and 53.76 milliseconds. Since the worst-case drift coefficient d has been fixed to 20 ppm and that the simulator drift follows a linear trend without noise, the n_{skip} values can be very easily deduced. Indeed we know that any device is subject to drift of a maximum of $20 \cdot 10^{-6} \cdot 128000 = 2.56$ milliseconds in the duration of a single beacon window. Therefore, we have in this case $n_{\text{skip}} = (\lfloor \delta_{\text{max}}/2.56 \rfloor - 1)$. Interestingly, δ_{max} values lower than 2.56 milliseconds can not be allowed because it is the drift experienced between the smallest possible interval between two beacon receptions. The selected δ_{max} then result in n_{skip} values of 0, 4, 10 and 20.

When focusing on the observed performances, Figure 5 also shows the good operation of the *LoRaSync* synchronization, since the slotted access allows to nearly double the maximum achievable throughput as expected. Regarding energy efficiency we find the result already presented in [46], which is that *Class A* performs better for low generation rates and the slotted access becomes preferable when traffic rate is higher than the threshold $\tau_{p/s1}$. With this network configuration, and if we consider 53.76 ms to be the upper

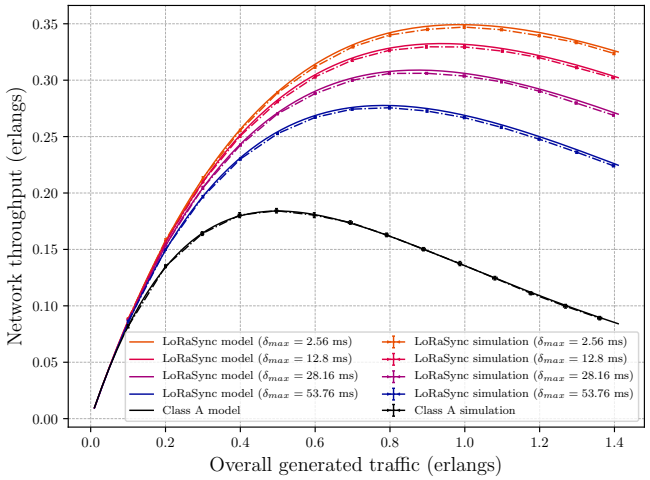


Fig. 5: Throughput model and simulation results

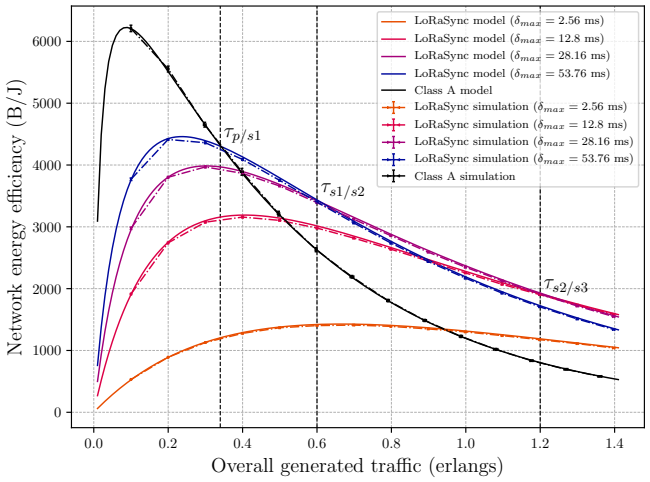


Fig. 6: Energy efficiency model and simulation results

bound to δ_{max}, $\tau_{p/s1}$ is equal to 0.34 erlangs. This value strongly depends on the hypothesis that devices are featured with a 20 ppm. drift, which was checked with our devices as we saw in Figure 3. Using devices of higher quality will naturally increase the Slotted ALOHA efficiency and move the crossing point towards a lower rate. More interestingly, Figure 6 shows that the value of δ_{max}, and therefore the slot size, has a strong impact on the network energy efficiency. Indeed, for a generated traffic range of 0.34 to 0.6 erlangs the maximum energy efficiency is attained when $\delta_{\text{max}} = 53.76$ ms ($n_{\text{skip}} = 20$). Then the $\delta_{\text{max}} = 28.16$ ms ($n_{\text{skip}} = 10$) curve is the best until 1.2 erlangs and finally $\delta_{\text{max}} = 12.8$ ms ($n_{\text{skip}} = 4$) prevails above that point. These intervals are represented by the $\tau_{s1/s2}$ and $\tau_{s2/s3}$ thresholds in Figure 6. We remark that the curve associated with $\delta_{\text{max}} = 2.56$ ms ($n_{\text{skip}} = 0$) is never optimal in terms of energy efficiency. This is consistent with the preliminary results presented in [14] that already showed that the beacon-skipping mechanism was unavoidable when

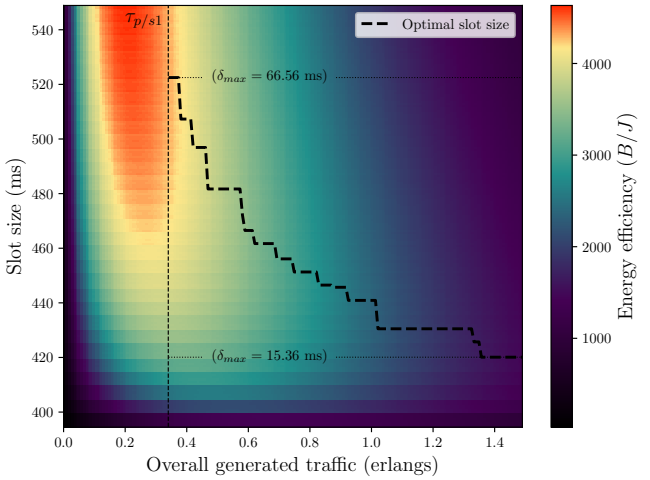


Fig. 7: Energy efficiency as a function of the slot size (2000 devices)

trying to optimize energy efficiency. We also notice that the optimal amount of skipped beacons decreases as the traffic load increases. In order to further explore this finding, the next part sheds some light on the relationship between slot size and energy efficiency.

B. Slot size optimization

This part aims at evaluating the impact of the *LoRaSync* slot size on the network performances. Indeed the slot margin length δ_{max} has an effect on the maximum achievable throughput, but also on the synchronization periodicity which in turn alters the device power consumption. This makes it a relevant parameter to consider when optimizing energy efficiency. In order to represent a realistic scenario, the models presented in the previous Section has been used to analyze the same large-scale deployment of 2000 devices.

From equation 4 we know that the slot size depends on δ_{max}, which represents the maximum device drift allowed. Increasing δ_{max} will result in larger slots, ultimately decreasing the number of slots per slotframe and the overall throughput. On the other hand the bigger δ_{max}, the more devices are able to drift. This means that they are able to skip more beacons. Each additional beacon skipped reduces the overall device power consumption. Energy efficiency, defined as the ratio between the throughput and power consumption, thus strongly depends on the slot size. It has been plot as a function of the slot size and generated traffic in Figure 7.

At each rate, the slot size associated with the maximum energy efficiency is represented by the black dashed line. The line is only traced for generation rates bigger than the $\tau_{p/s1}$ threshold introduced in Figure 6. Indeed the asynchronous Pure ALOHA access should be preferred in terms of energy efficiency for rates lower than this threshold, it is therefore useless to consider the slot size there. This line shows that the optimal slot size decreases as the network load increases. Remarkably, with the same maximum ToA, smaller slots mean

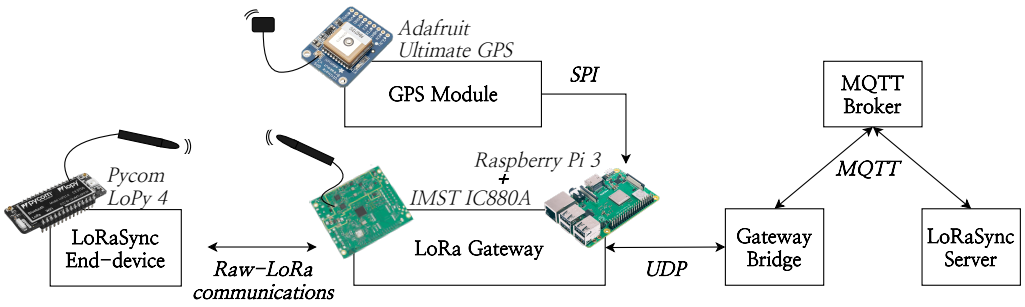


Fig. 8: Experimental testbed infrastructure.

that devices are allowed to drift of a smaller amount. As a result a shorter synchronization period is required, which implies consuming more power in beacon receptions. Such short margins are therefore only suitable for high transmission rates. Conversely, wide slots allow to space out synchronization events, but reduce the number of slots available per slotframe. This allows to save power at the cost of a reduced maximum achievable throughput, which is not a problem when the traffic load is low. In a nutshell the more traffic there is, the more the margins should be reduced.

All in all, this result shows that the slot size should ideally be adapted to the traffic load in order to maximize energy efficiency. We presented in [46] the TREMA mechanism to dynamically switch between Pure and Slotted ALOHA depending on the probed traffic conditions. Interestingly, this work could easily be extended to dynamically adapt the slot size to the generated load in order to further optimize the energy efficiency of the network.

VI. Proof of Concept on Real Hardware

In order to complement the previous theoretical analysis, *LoRaSync* was implemented on a small-scale research testbed. This proof-of-concept demonstrates that our mechanism is capable of bounding the synchronization error despite real hardware constraints. We additionally provide hands-on insights on how to handle the hardware and software challenges faced when synchronizing low-cost LoRa devices. Finally, we provide a performance evaluation of the Pure and Slotted ALOHA access schemes implemented on the testbed with 10 devices, which reveal a discrepancy between the models and reality. In that, the hardware implementation allows us to demonstrate the impact of capture effect on the channel throughput.

A. Architectural challenges

A LoRa testbed was set up according to the typical LPWAN architecture, *i.e.* a server, a gateway and devices, as depicted in Figure 8.

The server side mimicks the ChirpStack architecture, in which a MQTT broker centralizes the communications of all gateways. The ChirpStack gateway bridge allows to translate the MQTT streams into UDP datagrams understandable by the gateways. Finally, a custom *LoRaSync* server was developed to control and monitor the network with MQTT messages. The gateway is composed of a Raspberry Pi 3 running the Semtech Packet Forwarder software, used in conjunction with an IMST IC880A LoRa concentrator. The testbed devices are LoPy 4 chips developed by Pycom, offering quick prototyping capabilities for a relatively low cost.

By default, LoRa gateways are only capable of triggering downlink transmissions *(i)* immediately, or *(ii)* after a short delay following an uplink transmission (*Class A* RX windows). In order to allow timestamp-based downlink transmissions, an Adafruit Ultimate GPS chip was connected to the Raspberry Pi through Serial Peripheral Interface (SPI). It provides time and location information to the gateway program, and also sends a direct Pulse Per Second (PPS) signal to the concentrator enabling the triggering of transmissions with a 10 nanoseconds precision [47]. This GPS chip requires a direct line-of-sight with the sky, as a result our gateway has been setup outdoors. This module ultimately allows the periodic broadcasting of the *LoRaSync* synchronization beacons.

B. Software challenges

This Section details the firmware adjustments that had to be made to implement *LoRaSync* on LoPy 4 chips. It must be pointed out that this hardware that does not support *Class B*, and that a big portion of its behavior had first to be reproduced. The main challenge faced when programming *LoRaSync* on such devices was to schedule event executions with absolute timestamps, even though the embedded time library only allows callbacks after relative delays. To achieve this goal, the device logic was first split onto four threads, each assigned to a specific task: *(i)* message generation, *(ii)* frame reception, *(iii)* handling of received messages, and *(iv)* frame transmission. In this way, events requiring a precise timing such as frame transmissions and receptions may lock the CPU when needed, while the other tasks may proceed their execution freely during non-critical time.

Additionally, a software overlay was developed in order to handle critical method executions. Once synchronized, a node is able to estimate its current timestamp with its internal counter according to equation (2). When a timestamp event is scheduled, relative callbacks can then be used to trigger a call

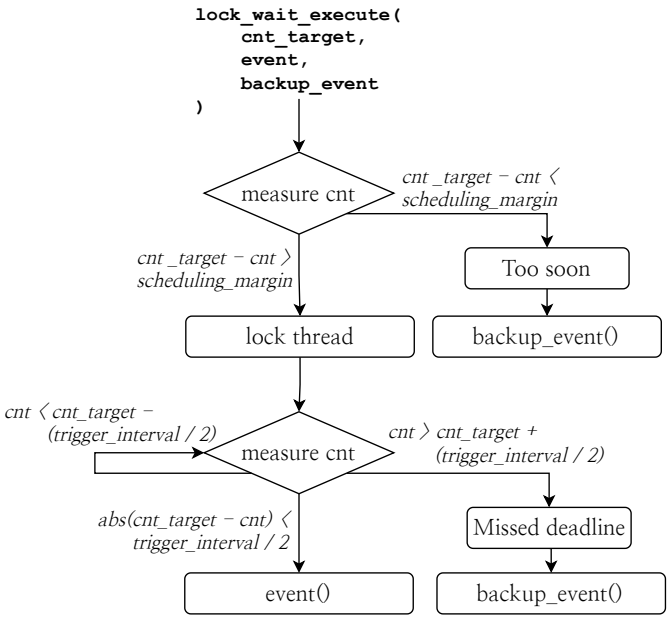


Fig. 9: `lock_wait_execute` flowchart.

to the `lock_wait_execute` method, capable of handling time-critical events. This method is in charge of locking the hardware resources for the current thread, and waiting until the internal counter falls within a precise confidence interval around the target timestamp (*c.f.* Figure 9). It then triggers the event execution with a ~ 100 microseconds precision. In the case where resources could not be locked in time for the critical method call, a backup event passed as parameter is executed in order to correct the failure.

Once the device is synchronized, this triggering process allows to track beacons by turning the radio on just before the next broadcast. It also enables uplink timestamp-based transmissions, which ultimately allows to implement the *Class S* timeslots.

C. Clock correction demonstration

In order to demonstrate the good operation of *LoRaSync*, the clock drift of a periodically synchronized device is plot in Figure 10. In order to facilitate the experiments, the node has been set indoor, and is not in direct line of sight with our outdoors gateway (*c.f.* Section VI-A). It notably ensures that the temperature is relatively constant for the duration of the experiment. This single device is setup with $\delta_{\max} = 39.16$ ms, a worst-case drift coefficient $d = 20$ ppm and a noise margin $\nu = 11$ ms. This setting results in $n_{\text{skip}} = 10$, meaning that a beacon is received every $11 \times 128 = 1408$ seconds. Beacon receptions are represented on the graph by black vertical dashed lines.

Here it is clear that the device never crosses the maximum drift allowed line $\delta_{\max} = 39.16$. The margin here appears particularly wide because the actual drift coefficient of the chosen device is $d = -1.5$ ppm, which is very small compared to the worst-case estimation of 20 ppm. The drift model

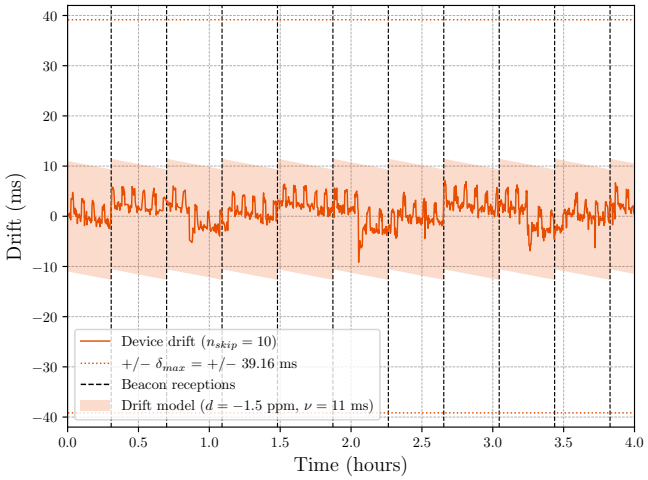


Fig. 10: Device drift measurements

interval has been plot over the actual drift in order to highlight the synchronization events. It shows that in this particular case, the noise plays a bigger role in clock errors than the linear component of the drift.

All in all, this proof of concept shows that the synchronization error is kept below the worst-case drift threshold $\delta_{\max}$. *LoRaSync* therefore operates as expected in a realistic context.

D. Impact of the Capture Effect on the testbed performances

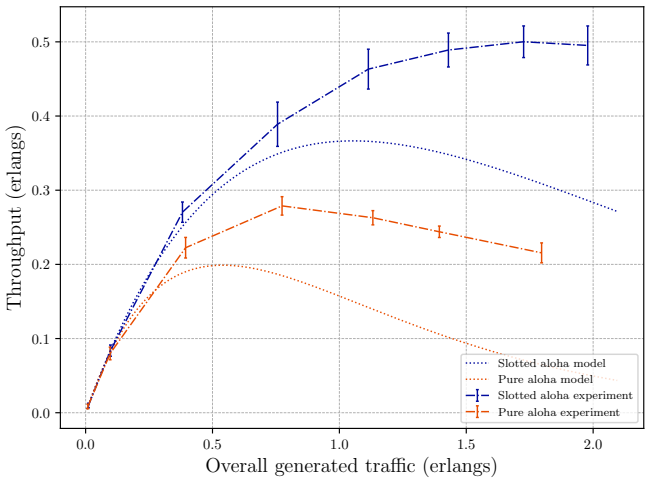


Fig. 11: Experimental pure and slotted ALOHA throughput

Additionally to this proof of concept with one device, we have evaluated the performances of the testbed under Pure and Slotted ALOHA access schemes for various generation rates with 10 devices. All devices are setup indoors at the same location. For the Slotted ALOHA scheme, we use LoRaSync parameterized with $\delta_{\max} = 39.16$, which results in the same parameters as in Section VI-C. For each generation rate the network runs for 30 minutes, and the throughput is sampled every minute. Errorbars represent 95% confidence intervals

computed with Student's t law. Results are plotted in Figure 11, along with ALOHA models presented in Section V for the sake of comparison. Here, it is obvious that the models underestimate the real-world throughput. As a result, we have thoroughly investigated this finding in a separate contribution [15], which presents experimental models able to accurately picture the testbed throughput. One of topics explored in this complementary study is that the impact of capture effect on the network heavily depends on the deployment layout. Even though we were able to find a model that applies to our experimental setup, more studies are still desired to generalize this result. Regardless, the increased performances experienced with the Slotted access prove the viability of the synchronization approach in a real deployment.

VII. Conclusion and perspectives

In this paper we introduced *LoRaSync*, a synchronization scheme designed for LoRa networks that aims at maximizing the device energy efficiency. Synchronization was used to setup a Slotted ALOHA access providing capacity improvements compared to the legacy Pure ALOHA scheme. *LoRaSync* slots may however be leveraged to implement any slotted MAC protocol. Synchronization is achieved through beacon broadcasts, and is therefore scalable to any network size. The *LoRaSync* slots contain margins to cope for the worst-case device clock drift, ensuring a reliable operation even on low-cost hardware. A beacon skipping mechanism is implemented so that devices only realign their internal clock when their current drift is likely to exceed the margin length. This way, the energy devices spend listening to incoming beacon broadcasts is minimized.

Throughput, power consumption and energy efficiency models were established to assess the performances of the Slotted ALOHA scheme built on top of *LoRaSync*. This access scheme was compared to the asynchronous Pure ALOHA access, and the LoRaWAN-Sim simulator was used to verify the consistency of the results. Such modeling ultimately allows to find the most efficient slot size for any traffic load. Results show that the more traffic is generated, the smaller margins should be in order to maximize energy efficiency at all times. *LoRaSync* was implemented on an experimental testbed, showing its capability to successfully correct the timing errors and keep the clock drift below the maximum threshold allowed even on a real-hardware setup. This proof-of-concept also reveals the impact of capture effect in a realistic setup, and this phenomenon has been studied in depth in a separate contribution [15].

All in all, *LoRaSync* provides a robust and energy efficient basis to allow development of slotted schemes over LoRaWAN. It must be pointed out that the simple Slotted ALOHA scheme used in this paper to demonstrate the viability of the mechanism is not flawless. For instance, a severe competition occurs during the first timeslot due to traffic accumulating during the beacon reserved and guard intervals. In that, more advanced schemes such as scheduling techniques should be developed, and *LoRaSync* has been designed as robust cornerstone able to support these future improvements. Besides, the TREMA switching mechanism allowing to select synchronous or asynchronous access schemes depending on the traffic load should be implemented, and improved to also adapt the slot size. Finally, all these contributions should be extended to operate in multi-gateway scenarios as well.

References

[1] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "A comparative study of LPWAN technologies for large-scale IoT deployment," *ICT Express*, vol. 5, no. 1, pp. 1–7, Mar. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405959517302953

[2] M. Rizzi, P. Ferrari, A. Flammini, and E. Sisinni, "Evaluation of the IoT LoRaWAN Solution for Distributed Measurement Applications," *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 12, pp. 3340–3349, Dec. 2017, conference Name: IEEE Transactions on Instrumentation and Measurement.

[3] K. Mikhaylov, J. Petäjäjärvi, and T. Haenninen, "Analysis of capacity and scalability of the lora low power wide area network technology," in *European Wireless 2016; 22th European Wireless Conference*, ser. Proceedings of EW '16, Oulu, Finland, May 2016, pp. 1–6.

[4] M. C. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do LoRa Low-Power Wide-Area Networks Scale?" in *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, ser. MSWiM '16. New York, NY, USA: ACM, 2016, pp. 59–67, event-place: Malta, Malta. [Online]. Available: http://doi.acm.org/10.1145/2988287.2989163

[5] D. Bankov, E. Khorov, and A. Lyakhov, "On the Limits of LoRaWAN Channel Access," in *2016 International Conference on Engineering and Telecommunication (EnT)*, Nov. 2016, pp. 10–14.

[6] O. Georgiou and U. Raza, "Low Power Wide Area Network Analysis: Can LoRa Scale?" *IEEE Wireless Communications Letters*, vol. 6, no. 2, pp. 162–165, Apr. 2017.

[7] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, "Understanding the Limits of LoRaWAN," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 34–40, Sep. 2017.

[8] F. Van den Abeele, J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Scalability Analysis of Large-Scale LoRaWAN Networks in ns-3," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2186–2198, Dec. 2017, conference Name: IEEE Internet of Things Journal.

[9] A. M. Yousuf, E. M. Rochester, B. Ousat, and M. Ghaderi, "Throughput, Coverage and Scalability of LoRa LPWAN for Internet of Things," in *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, Jun. 2018, pp. 1–10, iSSN: 1548-615X.

[10] N. Abramson, "THE ALOHA SYSTEM: another alternative for computer communications," in *Proc. of AFIPS '70 (Fall)*, ser. AFIPS '70 (Fall). Houston, Texas: ACM, Nov. 1970, pp. 281–285.

[11] T. Polonelli, D. Brunelli, A. Marzocchi, and L. Benini, "Slotted ALOHA on LoRaWAN-Design, Analysis, and Deployment," *Sensors*, vol. 19, no. 4, p. 838, Jan. 2019.

[12] J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Low Overhead Scheduling of LoRa Transmissions for Improved Scalability," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3097–3109, Apr. 2019.

[13] L. Roberts, "ALOHA Packet System With and Without Slots and Capture," *ACM SIGCOMM Computer Communication Review*, vol. 5, pp. 28–42, Apr. 1975.

[14] L. Chasserat, N. Accettura, and P. Berthou, "Short: Achieving energy efficiency in dense LoRaWANs through TDMA," in *IEEE WoWMoM 2020*, Cork, Ireland, Aug. 2020.

[15] ——, "Experimental throughput models for LoRa networks with capture effect," in *IEEE STWiMob 2022*, Thessaloniki, Greece, Oct. 2022. [Online]. Available: https://hal.archives-ouvertes.fr/hal-03766766

[16] S. El-Khamy, S. Shaaban, and E. Thabet, "Frequency-hopped multi-user chirp modulation (FH/M-CM) for multipath fading channels," in *Proceedings of the Sixteenth National Radio Science Conference. NRSC'99 (IEEE Cat. No.99EX249)*, Feb. 1999.

[17] M. Knight and B. Seeber, "Decoding LoRa: Realizing a Modern LPWAN with SDR," *Proc. of the GNU Radio Conference*, vol. 1, no. 1, Sep. 2016.

[18] L. Alliance, "LoRaWAN® L2 1.0.4 Specification (TS001-1.0.4)," Oct. 2020.

[19] R. Piyare, A. L. Murphy, M. Magno, and L. Benini, "On-Demand TDMA for Energy Efficient Data Collection with LoRa and Wake-up Receiver," in *2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Oct. 2018, pp. 1–4, iSSN: 2160-4886.

[20] L. Beltramelli, A. Mahmood, P. Osterberg, M. Gidlund, P. Ferrari, and E. Sisinni, "Energy efficiency of slotted lorawan communication with out-of-band synchronization," *IEEE Transactions on Instrumentation and Measurement*, p. 1–1, 2021.

[21] C. Garrido-Hidalgo, J. Haxhibeqiri, B. Moons, J. Hoebeke, T. Olivares, F. J. Ramirez, and A. Fernández-Caballero, "LoRaWAN Scheduling: From Concept to Implementation," *IEEE Internet of Things Journal*, 2021.

[22] R. K. Singh, R. Berkvens, and M. Weyn, "Synchronization and efficient channel hopping for power efficiency in LoRa networks: A comprehensive study," *Internet of Things*, vol. 11, p. 100233, Sep. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2542660520300664

[23] K. Q. Abdelfadeel, D. Zorbas, V. Cionca, B. O'Flynn, and D. Pesch, "FREE - Fine-grained Scheduling for Reliable and Energy Efficient Data Collection in LoRaWAN," *arXiv:1812.05744 [cs]*, Dec. 2018.

[24] B. Reynders, Q. Wang, P. Tuset-Peiro, X. Vilajosana, and S. Pollin, "Improving Reliability and Scalability of LoRaWANs Through Lightweight Scheduling," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1830–1842, Jun. 2018.

[25] D. Zorbas, K. Abdelfadeel, P. Kotzanikolaou, and D. Pesch, "TS-LoRa: Time-slotted LoRaWAN for the Industrial Internet of Things," *Computer Communications*, vol. 153, pp. 1–10, Mar. 2020.

[26] L. Tessaro, C. Raffaldi, M. Rossi, and D. Brunelli, "Lightweight Synchronization Algorithm with Self-Calibration for Industrial LORA Sensor Networks," in *2018 Workshop on Metrology for Industry 4.0 and IoT*, Apr. 2018, pp. 259–263.

[27] L. Kleinrock and F. Tobagi, "Random access techniques for data transmission over packet-switched radio channels," in *Proceedings of the May 19-22, 1975, national computer conference and exposition*, ser. AFIPS '75. Association for Computing Machinery, May 1975, pp. 187–201.

[28] S. Corporation, "AN1200.48 SX126x CAD Performance Evaluation," 2019.

[29] C. Pham, "Investigating and experimenting CSMA channel access mechanisms for LoRa IoT networks," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, Apr. 2018, pp. 1–6.

[30] ——, "Robust CSMA for long-range LoRa transmissions with image sensing devices," in *2018 Wireless Days (WD)*, Apr. 2018, pp. 116–122.

[31] S. Ahsan, S. A. Hassan, A. Adeel, and H. K. Qureshi, "Improving Channel Utilization of LoRaWAN by using Novel Channel Access Mechanism," in *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, Jun. 2019, pp. 1656–1661.

[32] M. O'Kennedy, T. Niesler, R. Wolhuter, and N. Mitton, "Practical evaluation of carrier sensing for a LoRa wildlife monitoring network," in *2020 IFIP Networking Conference (Networking)*, Jun. 2020, pp. 614–618.

[33] C. Pham and M. Ehsan, "Dense Deployment of LoRa Networks: Expectations and Limits of Channel Activity Detection and Capture Effect for Radio Channel Access," *Sensors*, vol. 21, no. 3, p. 825, Jan. 2021.

[34] F. Tobagi and L. Kleinrock, "Packet Switching in Radio Channels: Part II - The Hidden Terminal Problem in Carrier Sense Multiple-Access and the Busy-Tone Solution," *IEEE Transactions on Communications*, vol. 23, no. 12, pp. 1417–1433, Dec. 1975.

[35] A.-I. Pop, U. Raza, P. Kulkarni, and M. Sooriyabandara, "Does Bidirectional Traffic Do More Harm Than Good in LoRaWAN Based LPWA Networks?" *arXiv:1704.04174 [cs]*, Dec. 2017.

[36] V. Di Vincenzo, M. Heusse, and B. Tourancheau, "Improving Downlink Scalability in LoRaWAN," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, May 2019, pp. 1–7, iSSN: 1938-1883.

[37] D. Veitch, S. Babu, and A. Pàsztor, "Robust synchronization of software clocks across the internet," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, ser. IMC '04. New York, NY, USA: Association for Computing Machinery, Oct. 2004, pp. 219–232. [Online]. Available: https://doi.org/10.1145/1028788.1028817

[38] A. Mallat and L. Vandendorpe, "Joint estimation of the time delay and the clock drift and offset using UWB signals," in *2014 IEEE International Conference on Communications (ICC)*, Jun. 2014, pp. 5474–5480, iSSN: 1938-1883.

[39] Semtech Corporation, "An1200.22 lora modulation basics application note," 2015.

[40] D. Croce, M. Gucciardo, I. Tinnirello, D. Garlisi, and S. Mangione, "Impact of Spreading Factor Imperfect Orthogonality in LoRa Communications," in *Digital Communication. Towards a Smart and Secure Future Internet*, ser. Communications in Computer and Information Science, A. Piva, I. Tinnirello, and S. Morosi, Eds. Cham: Springer International Publishing, 2017, pp. 165–179.

[41] LoRa Alliance, "LoRaWAN Regional parameters v1.1," 2017.

[42] U. Noreen, L. Clavier, and A. Bounceur, "LoRa-like CSS-based PHY layer, Capture Effect and Serial Interference Cancellation," in *European Wireless 2018; 24th European Wireless Conference*, May 2018, pp. 1–6.

[43] R. Rom and M. Sidi, "Multiple Access Protocols: Performance and Analysis," *SIGMETRICS Perform. Evaluation Rev.*, vol. 18, p. 11, 1991.

[44] A. Brand and H. Aghvami, *Multiple Access Protocols for Mobile Communications*. John Wiley & Sons Ltd, Apr. 2002.

[45] Semtech Corporation, "SX1276/77/78/79 datasheet," Jan. 2019.

[46] L. Chasserat, N. Accettura, B. Prabhu, and P. Berthou, "TREMA: A traffic-aware energy efficient MAC protocol to adapt the LoRaWAN capacity," in *2021 International Conference on Computer Communications and Networks (ICCCN)*, Jul. 2021, pp. 1–6, iSSN: 2637-9430.

[47] A. Industries, "Adafruit Ultimate GPS Breakout v3 datasheet," 2021.